```
/* Alice Routine
 /    - Receives keypad input and displays on LCD.
 /    - Controls program flow in Oscar and Bob
 /    - Encryption routine
 /    - Ben Anderson
 */

#include <pic.h>
#include <stdio.h>
#include <string.h>
#include "lcd.h"
#include "defs.h"
#include "Keypad.h"
#include <math.h>
#include <stdlib.h>
#include <time.h>

#define  NEXTPAGE RC4

bank1 const char * str;
bank1 char showNum[8];
bit next,done = 0;

//Deepak's Variables
bank1 int flag=1;
bank1 unsigned int prime_p,   // First Prime
          prime_q,            // Second Prime
          n,                  // n = prime_p * prime_q
          phi,                // phi = (prime p -1)*(prime q-1)
          e,                  // e is relatively prime to and less than phi
          d,                  // d is the multiplicative inverse of e
          ciphertext,
          i = 0;              // standard incrementor

bank2 unsigned long temp,g,message = 0;   // This is the message to be
encrypted.



// Subroutines
int user_input(void);
int modInverse(int a,int p);

void main (void){

      TRISA = 0xFF;
       TRISB = 0;
      TRISC = 0;
      ADCON1 = 6;
      Wait_ms(100);

      lcd_init();             // initialize the LCD
      Wait_ms(100);

/*-----------------------------
 *    Welcome Message
 *-----------------------------
 */
      str = "   Welcome to   \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "   Kid Krypto   \0";
      lcd_puts(str);
```

```
      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    TrapDoor Introduction
 *-----------------------------
 *

      str = "Trap door functs\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "are those that  \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    TrapDoor Introduction II
 *-----------------------------
 */
      str = " RSA encryption \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = " uses two large \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
 *    TrapDoor Introduction III
 *--------------------------------
 */
      str = "It is very hard \0";
      lcd_puts(str);
```

```
      lcd_goto(0x40);
      str = "to factorize a  \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-------------------------------
 *    TrapDoor Introduction IV
 *-------------------------------
 */
      str = "When the primes \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "are sufficiently\0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-------------------------------
 *    TrapDoor Introduction V
 *-------------------------------
 */
      str = "For this example\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "we will use more\0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Input Prime p
 *-----------------------------
 */
```

```
      str = "We need a prime \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = " number p...    \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      next = 0;
      lcd_clear();
      Wait_ms(250);

      prime_p = user_input();

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

/*-----------------------------
 *    Input Prime q
 *-----------------------------
 */
      str = "We need a prime \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = " number q...    \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 0;
      next = 0;
      lcd_clear();
      Wait_ms(250);

      prime_q = user_input();

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

/*-----------------------------
 *    Calculate n
 *-----------------------------
 */
      n = prime_p*prime_q;

      str = "Calculate n=p*q \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "n=              \0";
      lcd_puts(str);
      lcd_goto(0x42);
      sprintf(showNum,"%u",n);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }
```

```
      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Input Secret Message
 *-----------------------------
 */
      while(!next){
         str = "Now we need a   \0";
         lcd_puts(str);
         lcd_goto(0x40);
         str = "secret message. \0";
         lcd puts(str);
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

      message = (unsigned long)(user_input());

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

/*-----------------------------
 *    Calculate Phi
 *-----------------------------
 */
      phi = (prime_p-1)*(prime_q-1);

      str = "Calculate phi   \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "phi=(p-1)*(q-1) \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
```

```
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

      str = "Phi=            \0";
      lcd_puts(str);
      lcd_goto(0x04);
      sprintf(showNum,"%u",phi);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Select e
 *-----------------------------
 */
      str = "Now we choose a \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "value e= 3->7   \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      Wait_ms(250);


      e = user_input();

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);


/*-----------------------------
 *    Determine d
 *-----------------------------
 */
      d = modInverse(e,phi);

      str = "Now we must find\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "the value d,    \0";
      lcd_puts(str);
```

```
      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

      str = "d=              \0";
      lcd_puts(str);
      lcd_goto(0x02);
      sprintf(showNum,"%u",d);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      lcd_goto(0x40);
      str = "d satisfies the \0";

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Key Generation Review
 *-----------------------------
 */

      str = "What have we    \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "done with all   \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Key Generation Review II
 *-----------------------------
 */

      str = "RSA uses two key\0";
```

```
      lcd_puts(str);
      lcd_goto(0x40);
      str = "structures:     \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
 *    Key Generation Review III
 *--------------------------------
 *

      str = "RSA uses two key\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "structures:     \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
 *    Key Generation Review IV
 *--------------------------------
 */

      str = "We generate both\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "keys first.     \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
```

```
 *    Key Generation Review V
 *--------------------------------
 */

      str = "Anyone with our \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "public key can  \0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Public Key
 *-----------------------------
 */
      str = "e=              \0";
      lcd_puts(str);
      lcd_goto(0x02);
      sprintf(showNum,"%u",e);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      lcd_goto(0x40);
      str = "n=              \0";
      lcd_puts(str);
      lcd_goto(0x42);
      sprintf(showNum,"%u",n);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Private Key
 *-----------------------------
 */
      str = "d=              \0";
      lcd_puts(str);
      lcd_goto(0x02);
      sprintf(showNum,"%u",d);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;
```

```
      lcd_goto(0x40);
      str = "n=              \0";
      lcd_puts(str);
      lcd_goto(0x42);
      sprintf(showNum,"%u",n);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
 *    Encrypting Plaintext
 *--------------------------------
 *

      str = "Ok, great! But  \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "how can we use a\0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*--------------------------------
 *    Encrypting Plaintext II
 *--------------------------------
 *

      str = "Using this eqtn.\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "we can calculate\0";
      lcd_puts(str);

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
```

```
      Wait_ms(250);

/*--------------------------------
 *    Encrypting Plaintext III
 *--------------------------------
 *

      str = "In our case:    \0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "C=    mod       \0";
      lcd_puts(str);

      lcd_goto(0x42);
      sprintf(showNum,"%u",message);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      lcd_goto(0x43);
      lcd_putch(0x5E);

      lcd_goto(0x44);
      sprintf(showNum,"%u",e);
      lcd_puts(showNum);
      for(i=0;i<8;i++) showNum[i]=0;

      lcd_goto(0x4A);
      sprintf(showNum,"%u",n);
      lcd puts(showNum);
      for(i=0;i<16;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

/*-----------------------------
 *    Calculate Cyphertext
 *-----------------------------
 */
      g = 1;

      for(i=0;i<e;i++){
         g = (unsigned long)g*(unsigned long)message;
      }


      str = "message^e:      \0";
      lcd_puts(str);
      lcd_goto(0x40);

      sprintf(showNum,"%9u",(unsigned long)g);
      lcd_puts(showNum);
      for(i=0;i<16;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
```

```
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);

      ciphertext = g%n;

      str = "Calculate Cipher\0";
      lcd_puts(str);
      lcd_goto(0x40);
      str = "Cipher=         \0";
      lcd_puts(str);

      lcd_goto(0x47);

      sprintf(showNum,"%u",ciphertext);
      lcd_puts(showNum);
      for(i=0;i<16;i++) showNum[i]=0;

      while(!next){
         next = kpd_dataReady();
      }

      NEXTPAGE = 1;
      Wait_ms(25);
      NEXTPAGE = 0;

      next = 0;
      lcd_clear();
      Wait_ms(250);


      while(!next){
         next = kpd_dataReady();
      }

      lcd_clear();
      Wait_ms(500);
} // End of file


/*
 * User Input
 */
int user_input(void){

      unsigned char cnt = 0;
      unsigned char dec = 1;
      unsigned int  num = 0;
      unsigned char entry = 0;
      const char * str;
      unsigned char buf[3];
      char display[5];
      unsigned char i = 0;
      unsigned char size = 0;

      lcd_clear();
      str = "Enter number: \0";
      lcd_puts(str);
```

```
      lcd_goto(0x40);

      while(!done){           // Program moves forward when 'Enter' is pressed
         if(kpd_dataReady()){
            Wait_ms(250);
            entry = kpd_getChar();
            Wait_ms(10);
            if(entry == 'C'){ // Clear entry
               entry = 0;
               done = 0;
               cnt = 0;
               lcd_clear();
               str = "Enter number: \0";
               lcd_puts(str);
               lcd_goto(0x40);
               for(i=0;i<3;i++) buf[i] = 0;
            }
            else if(entry == 'E'){  // Enter is pressed
               entry = 0;
               done = 1;
               lcd_clear();
               Wait_ms(250);
            }
            else{                // Another Key was pressed
               lcd_putch(entry);
               if(cnt<3){        // Input Verification
                  buf[cnt] = entry;
                  cnt++;
                  entry = 0;
               }
               else{          // Input Error
                  lcd_clear();
                  str = "Enter number: \0";
                  lcd_puts(str);
                  lcd_goto(0x40);
                  str = "Too Big Re-enter\0";
                  lcd_puts(str);
                  Wait_ms(1000);

                  entry = 0;
                  done = 0;
                  cnt = 0;
                  lcd_clear();
                  str = "Enter number: \0";
                  lcd_puts(str);
                  lcd_goto(0x40);
                  for(i=0;i<3;i++) buf[i] = 0;
               }
               entry = 0;
            } // End Another Key was pressed
         } // End if kpd_pressed
      } // End while(!done)

      done = 0;

      // Convert input to integer.
      lcd_clear();
/*
      str = "Convert to int  \0";
      lcd_puts(str);
      lcd_goto(0x40);
      Wait_ms(2000);
      lcd_clear();
*/
```

```
      num = atoi(buf);

/*    size = sprintf(display,"%d",num);

      lcd_clear();
      lcd_puts(display);
      Wait_ms(2000);*/

      lcd_clear();
      cnt = 0;
      dec = 1;
      entry = 0;
      done = 0;

      return num;
} // End of user_input();

/*
 * Modular Inversion. Direct implementation using Euclid's alg.
 * Loop unrolled to avoid swapping variables.
 *
 * From: IDEA sources in "Applied Cryptography" by Bruce Schneier (p. 522)
 *
 * Input:
 * a,p   Two prng_num
 *
 * Output:
 * prng_num which is the inverse of a modulo p
 *
 */
int modInverse(int a,int p)
{
   int q,y,t0,t1;
   #ifdef ITER_COUNT
   int count = 0;
   #endif

   if (a <= 1)       // trivial cases
      return(a);

   #ifdef ITER_COUNT
   count++;
   #endif
   t1 = p / a;
   y  = p % a;
   if (y ==1)
      return(p - t1);

   t0 = 1;

   do{
      #ifdef ITER_COUNT
      count++;
      #endif
       q = a / y;
       a = a % y;
       t0 += q * t1;

      if (a == 1){
         #ifdef ITER_COUNT
         iterations[count]++;
         #endif
         return(t0);
      }
```

```
      #ifdef ITER_COUNT
      count++;
      #endif

       q = y / a;
      y = y % a;
       t1 += q * t0;
   }
   while (y != 1);

   #ifdef ITER_COUNT
   iterations[count]++;
   #endif
   return(p - t1);

}
```